UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/748,285	12/29/2003	Stephan J. Jourdan	Intel 2207/17048	8395

7590 07/30/2007

KENYON & KENYON
Suite 600
333 W. San Carlos Street
San Jose, CA 95110-2711

EXAMINER	
MEONSKE, TONIA L	

ART UNIT	PAPER NUMBER
2181	

MAIL DATE	DELIVERY MODE
07/30/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/748,285	JOURDAN, STEPHAN J.	
	Examiner	Art Unit	
	Vincent Lai	2181	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 June 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Response to Arguments

1. Applicant's arguments filed 4 June 2007 have been fully considered but they are not persuasive.

Applicant had argued the "The Rotenberg reference is not an enabling reference as it relates to path associativity because although it mentions path associativity as being a possible aspect of a future embodiment: it is silent regarding how to implement it into a trace cache, and none of the additional references cited cure this deficiency. The Examiner has, therefore, failed to make a prima facie case of obviousness."

Applicant admits that Rotenberg teaches path associativity. Examiner argues that one having ordinary skill in the art would be able to implement path associativity into a trace cache. Applicant has not made specific arguments pertaining to why one having ordinary skill in the art would not be able to implement path associativity. Without such arguments, Examiner cannot respond and is not persuaded by such arguments.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1-2, 11-12, and 16-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg et al ("Trace Cache: a Low Latency Approach to High Bandwidth Instruction Fetching", Eric Rotenberg, Steve Bennett, James E Smith, IEEE, 1996; herein referred to as "Rotenberg".).

Regarding **independent claim 1**, Rotenberg discloses *a method comprising: reviewing a first branching behavior of a first previous set of branching instructions executed by a processor* [see Rotenberg, Page. 5, Col. 2, lines 16-18; Examiner's note: Rotenberg discloses comparing predictions made by a branch predictor with the branching behavior of a trace, thus reviewing the branch behavior.]; *reviewing multiple traces* [see Rotenberg, Page 5, Col. 2, lines 21-23 "...the fetch address matches the tag..."; Page 5, Col. 1, line 18 "tag: identifies the starting address of the trace."; Page. 6, Col. 1, lines 27-33;]; *and selecting a trace from among the multiple traces based on the branching behavior of the first previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 16-18 "The predictor generates multiple branch predictions while the caches are accessed."; Page 5, Col. 2, lines 21-23; "...the branch predictions match the branch flags..."; Page 5, Col. 1, lines 19-21 "branch flags: there is a single bit for

each branch within the trace to indicate the path followed after the branch (taken/not taken).].

Whereas Rotenberg does teach the reviewing of multiple traces (See above rejection), Rotenberg does not teach that the multiple traces have a same beginning instruction.

Rotenberg does teach how such features would be advantageous (See page 6, lines 27-33).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Rotenberg to include multiple traces that have a same beginning instruction. Rotenberg already teaches that the implementation used is mean to be the simplest possible and that one may desire advanced features such as being able to have multiple traces that have a same beginning instruction. One having ordinary skill in the art would recognize the advantageous of using such a feature and would be able to implement it.

Regarding **claim 2**, Rotenberg discloses *the method of claim 1, further comprising: selecting the trace from among the multiple traces that has a second branching behavior of a second previous set of branching instructions that matches the first branching behavior of the first previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 20-22, "...the branch predictions match the branch flags..."; Examiner's note: Inherently, a trace may be accessed multiple times, thus reviewing a second branching behavior and comparing it to a prior behavior.].

Regarding **independent claim 11**, Rotenberg discloses *a processor comprising: a branch predictor* [see Rotenberg, Page 4, Fig. 3, element "Multiple Branch Predictor"] *to review a first branching behavior of a first previous set of branching instructions executed by a processor* [see Rotenberg, Page 5, Col. 2, lines 21-23 "...the fetch address matches the tag..."; Page 5, Col. 1, line 18 "tag: identifies the starting address of the trace."; Page. 6, Col. 1, lines 27-33;]*; a trace cache* [see Rotenberg, Page 5, Fig. 4] *to store multiple traces* [see Rotenberg, Page 5, Col. 2, lines 21-23 "...the fetch address matches the tag..."; Page 5, Col. 1, line 18 "tag: identifies the starting address of the trace."]*; and a fetching mechanism to retrieve a trace from among the multiple traces based on the first branching behavior of the previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 16-18 "The predictor generates multiple branch predictions while the caches are accessed."; Page 5, Col. 2, lines 21-23; "...the branch predictions match the branch flags..."; Page 5, Col. 1, lines 19-21 "branch flags: there is a single bit for each branch within the trace to indicate the path followed after the branch (taken/not taken).].

Whereas Rotenberg does teach the reviewing of multiple traces (See above rejection), Rotenberg does not teach that the multiple traces have a same beginning instruction.

Rotenberg does teach how such features would be advantageous (See page 6, lines 27-33).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Rotenberg to include multiple traces that have a same beginning instruction. Rotenberg already teaches that the implementation used is mean to be the simplest possible and that one may desire advanced features such as being able to have multiple traces that have a same beginning instruction. One having ordinary skill in the art would recognize the advantageous of using such a feature and would be able to implement it.

Regarding **claim 12**, Rotenberg discloses *the processor of claim 11, wherein the fetching mechanism is to select the trace from among the multiple traces that has a second branching behavior of a second previous set of branching instructions that matches the first branching behavior of the first previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 20-22, "...the branch predictions match the branch flags..."].

Regarding **independent claim 16**, Rotenberg *a system comprising: a memory to store a set of instructions* [see Rotenberg, Page 1, Col. 1, lines 17-19; Examiner's note: Rotenberg discloses a trace cache within a superscalar environment. It would have been obvious to one of ordinary skill in the art at the time of invention that a processing environment would need a memory to store a set of instructions to provide functionality for the invention.]; *a processor coupled to the memory to execute the set of instructions* [see Rotenberg, Page. 1, Fig. 1; Examiner's note: It is inherent that Rotenberg intends

the trace cache to be utilized within a processor capable of executing instructions.], *the processor with a branch predictor* [see Rotenberg, Page 4, Fig. 3, element "Multiple Branch Predictor"] *to review a first branching behavior of a first previous set of branching instructions executed by a processor* [see Rotenberg, Page. 5, Col. 2, lines 16-18; Examiner's note: Rotenberg discloses comparing predictions made by a branch predictor with the branching behavior of a trace, thus reviewing the branch behavior.], *a trace cache* [see Rotenberg, Page 5, Fig. 4] *to store multiple traces* [see Rotenberg, Page 5, Col. 2, lines 21-23 "...the fetch address matches the tag..."; Page 5, Col. 1, line 18 "tag: identifies the starting address of the trace."; Page. 6, Col. 1, lines 27-33;], *and a fetching mechanism to retrieve a trace from among the multiple traces based on the first branching behavior of the previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 16-18 "The predictor generates multiple branch predictions while the caches are accessed."; Page 5, Col. 2, lines 21-23; "...the branch predictions match the branch flags..."; Page 5, Col. 1, lines 19-21 "branch flags: there is a single bit for each branch within the trace to indicate the path followed after the branch (taken/not taken).].

Whereas Rotenberg does teach the reviewing of multiple traces (See above rejection), Rotenberg does not teach that the multiple traces have a same beginning instruction.

Rotenberg does teach how such features would be advantageous (See page 6, lines 27-33).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Rotenberg to include multiple traces that have

a same beginning instruction. Rotenberg already teaches that the implementation used is mean to be the simplest possible and that one may desire advanced features such as being able to have multiple traces that have a same beginning instruction. One having ordinary skill in the art would recognize the advantageous of using such a feature and would be able to implement it.

Regarding **claim 17**, Rotenberg discloses *a system of claim 16, wherein the fetching mechanism is to select the trace from among the multiple traces that has a second branching behavior of a second previous set of branching instructions that matches the first branching behavior of the first previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 20-22, "...the branch predictions match the branch flags..."].

3. Claims 3, 13, and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg in view of Nair (US Pat. No. 6,304,962).

Regarding **claim 3**, Rotenberg discloses the limitations as stated in **independent claim 1**.

Rotenberg does not explicitly disclose *generating a new trace if a divergence occurs in a pre-determined location in the trace.*

Nair does disclose *generating a new trace if a divergence occurs in a pre-determined location in the trace* [see Nair, Col. 8, lines 34-39].

The advantage of generating a new trace if a divergence occurs in a trace would have been to allow for a processor to dynamically update a trace cache with up-to-date information regarding the outcome of branches within a trace. Said updating would allow a processor to maintain an accurate representation of the most recent performance of a given trace. This advantage is desirable in the environment provided by Rotenberg as it would have allowed for fewer mispredictions of branch instructions while enabling a processor to maintain the use of a trace cache architecture. This advantage would have motivated one of ordinary skill in the art to allow a trace cache to be updated upon an invalid branch prediction of an instruction within a trace.

Regarding **claim 13**, Rotenberg discloses the limitations as stated in **independent claim 11**.

Rotenberg does not explicitly disclose *a processing core to* [executing] *the trace and to generate a new trace if a divergence occurs in a pre-determined location in the trace.*

Nair does disclose *a processing core to* [executing] *the trace and to generate a new trace if a divergence occurs in a pre-determined location in the trace* [see Nair, Col. 8, lines 34-39].

The advantage of generating a new trace if a divergence occurs in a trace would have been to allow for a processor to dynamically update a trace cache with up-to-date information regarding the outcome of branches within a trace. Said updating would allow a processor to maintain an accurate representation of the most recent

performance of a given trace. This advantage is desirable in the environment provided by Rotenberg as it would have allowed for fewer mispredictions of branch instructions while enabling a processor to maintain the use of a trace cache architecture. This advantage would have motivated one of ordinary skill in the art to allow a trace cache to be updated upon an invalid branch prediction of an instruction within a trace.

Regarding **claim 18**, Rotenberg discloses the limitations as stated in **independent claim 16**.

Rotenberg does not explicitly disclo*se a processing core to execute the trace and to generate a new trace if a divergence occurs in a pre-determined location in the trace.*

Nair does disclose *a processing core to execute the trace and to generate a new trace if a divergence occurs in a pre-determined location in the trace* [see Nair, Col. 8, lines 34-39].

The advantage of generating a new trace if a divergence occurs in a trace would have been to allow for a processor to dynamically update a trace cache with up-to-date information regarding the outcome of branches within a trace. Said updating would allow a processor to maintain an accurate representation of the most recent performance of a given trace. This advantage is desirable in the environment provided by Rotenberg as it would have allowed for fewer mispredictions of branch instructions while enabling a processor to maintain the use of a trace cache architecture. This advantage would have motivated one of ordinary skill in the art to allow a trace cache to be updated upon an invalid branch prediction of an instruction within a trace.

4. Claims 4-5, 14-15, and 19-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg in view of Nair in view of Patel (Trace Cache Design for Wide-Issue Superscalar Processors).

Regarding **claim 4**, Rotenberg and Nair disclose the limitations as stated in **claim 3**.

Rotenberg and Nair do not disclose *further comprising determining based on which instruction within a block of instructions creates the branch whether the new trace is generated.*

Patel does disclose *further comprising determining based on which instruction within a block of instructions creates the branch whether the new trace is generated* [see Patel, Page. 75, lines 7-9].

The advantage of determining whether a new trace should be generated based on its location within a block of instructions would have been to minimize the negative effects of trace fragmentation within the trace cache [see Patel, Page 75, lines 12-13], for example, in tight loop situations utilizing backwards branch instructions [see Patel, Page 75, lines 13-14]. This advantage is desirable in the environment disclosed by Rotenberg and Nair as it would have increased the throughput of the processor and the speed at which instructions could be fetched due to the minimization of fragmentation within the trace cache. This advantage would have motivated one of ordinary skill in the art at the time of invention to base the generation of new traces on the position of an

instruction, as disclosed by Patel, within the invention disclosed by Rotenberg and Patel.

Regarding **claim 5**, Rotenberg and Nair disclose the limitations as stated in **claim 3**.

Rotenberg and Nair do not explicitly disclose *further comprising determining, based on which block of instructions the branch occurs in, whether an alternate trace is generated.*

However, Patel discloses selectively packing traces dependent on their position within a trace block. Patel does not explicitly disclose selecting traces to be packed based on block position, however, Patel states the goal of selectively generating traces based on instruction position would have been to increase the fetch bandwidth of a trace cache. It would have been obvious to one of ordinary skill in the art at the time of invention to base the generation of a trace based on the block the trace resides in as it is merely a higher level of hierarchy (instruction – trace – block – cache) and the implementation would have been theoretically similar to that of denying the creation of a trace based on the instruction placement within a block. The advantages of selectively generating traces has been discussed in the arguments concerning the preceding claim and will not be repeated in this rejection. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to utilize the trace creation selection mechanism disclosed by Patel in a higher level of cache hierarchy, such as the block level.

Regarding **claim 14**, Rotenberg and Nair disclose the limitations as stated in **claim 13**.

Rotenberg and Nair do not disclose *the new trace* [being] *generated is based on which instruction within a block of instructions creates the branch.*

Patel does disclose *the new trace* [being] *generated is based on which instruction within a block of instructions creates the branch* [see Patel, Page. 75, lines 7-9].

The advantage of determining whether a new trace should be generated based on its location within a block of instructions would have been to minimize the negative effects of trace fragmentation within the trace cache [see Patel, Page 75, lines 12-13], for example, in tight loop situations utilizing backwards branch instructions [see Patel, Page 75, lines 13-14]. This advantage is desirable in the environment disclosed by Rotenberg and Nair as it would have increased the throughput of the processor and the speed at which instructions could be fetched due to the minimization of fragmentation within the trace cache. This advantage would have motivated one of ordinary skill in the art at the time of invention to base the generation of new traces on the position of an instruction, as disclosed by Patel, within the invention disclosed by Rotenberg and Patel.

Regarding **claim 15**, Rotenberg and Nair disclose the limitations as stated in **claim 13**.

Rotenberg and Nair do not explicitly disclose *an alternate trace* [being] *generated* [being] *based on which block of instructions the branch occurs in.*

However, Patel discloses selectively packing traces dependent on their position within a trace block. Patel does not explicitly disclose selecting traces to be packed based on block position, however, Patel states the goal of selectively generating traces based on instruction position would have been to increase the fetch bandwidth of a trace cache. It would have been obvious to one of ordinary skill in the art at the time of invention to base the generation of a trace based on the block the trace resides in as it is merely a higher level of hierarchy (instruction – trace – block – cache) and the implementation would have been theoretically similar to that of denying the creation of a trace based on the instruction placement within a block. The advantages of selectively generating traces has been discussed in the arguments concerning the preceding claim and will not be repeated in this rejection. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to utilize the trace creation selection mechanism disclosed by Patel in a higher level of cache hierarchy, such as the block level.

Regarding **claim 19**, Rotenberg and Nair disclose the limitations as stated in **claim 18**.

Rotenberg and Nair do not disclose *the new trace* [being] *generated is based on which instruction within a block of instructions creates the branch.*

Patel does disclose *the new trace* [being] *generated is based on which instruction within a block of instructions creates the branch* [see Patel, Page. 75, lines 7-9].

The advantage of determining whether a new trace should be generated based on its location within a block of instructions would have been to minimize the negative effects of trace fragmentation within the trace cache [see Patel, Page 75, lines 12-13], for example, in tight loop situations utilizing backwards branch instructions [see Patel, Page 75, lines 13-14]. This advantage is desirable in the environment disclosed by Rotenberg and Nair as it would have increased the throughput of the processor and the speed at which instructions could be fetched due to the minimization of fragmentation within the trace cache. This advantage would have motivated one of ordinary skill in the art at the time of invention to base the generation of new traces on the position of an instruction, as disclosed by Patel, within the invention disclosed by Rotenberg and Patel.

Regarding **claim 20**, Rotenberg and Nair disclose the limitations as stated in **claim 18**.

Rotenberg and Nair do not explicitly disclose an *alternate trace* [being] *generated* [being] *based on which block of instructions the branch occurs in.*

However, Patel discloses selectively packing traces dependent on their position within a trace block. Patel does not explicitly disclose selecting traces to be packed based on block position, however, Patel states the goal of selectively generating traces

based on instruction position would have been to increase the fetch bandwidth of a trace cache. It would have been obvious to one of ordinary skill in the art at the time of invention to base the generation of a trace based on the block the trace resides in as it is merely a higher level of hierarchy (instruction – trace – block – cache) and the implementation would have been theoretically similar to that of denying the creation of a trace based on the instruction placement within a block. The advantages of selectively generating traces has been discussed in the arguments concerning the preceding claim and will not be repeated in this rejection. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to utilize the trace creation selection mechanism disclosed by Patel in a higher level of cache hierarchy, such as the block level.

5. Claims 6-7 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg in view of Tanenbaum (Andrew S. Tanenbaum. *Structured Computer Organization,* 1984. Pg. 10-11; herein referred to as "Tanenbaum").

Regarding **independent claim 6**, Rotenberg discloses *reviewing a first branching behavior of a first previous set of branching instructions executed by a processor; reviewing multiple traces that have a same beginning instruction; and selecting a trace from among the multiple traces based on the branching behavior of the first previous set of branching instructions.*

Rotenberg does not disclose *a set of instructions residing in a storage medium, said set of instructions capable of being executed by a processor to implement a method*

However, Tanenbaum discloses that "hardware and software are logically equivalent" and that any hardware apparatus can be simulated in software [see Tanenbaum, p. 11, lines 11-13]. The advantage of implementing the method disclosed within claim 18 within a machine-accessible medium would have been to exploit the advantages of software-based approaches such as cost or ease of upgrading [see Tanenbaum, p. 11, lines 13-15]. This advantage would have motivated one of ordinary skill in the art to implement the method disclosed in the body of claim 6 in software as opposed to in hardware.

Regarding **claim 7**, Rotenberg and Tanenbaum disclose the limitations as stated in **independent claim 6**.

Rotenberg further discloses *selecting the trace from among the multiple traces that has a second branching behavior of a second previous set of branching instructions that matches the first branching behavior of the first previous set of branching instructions* [see Rotenberg, Page 5, Col. 2, lines 20-22, "...the branch predictions match the branch flags..."].

6. Claim 8 is rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg in view of Nair and in view of Tanenbaum.

Regarding **claim 8**, Rotenberg and Tanenbaum disclose the limitations as stated in **independent claim 6**.

Rotenberg and Tanenbaum do not explicitly disclose *generating a new trace if a divergence occurs in a pre-determined location in the trace.*

Nair does disclose *generating a new trace if a divergence occurs in a pre-determined location in the trace* [see Nair, Col. 8, lines 34-39].

The advantage of generating a new trace if a divergence occurs in a trace would have been to allow for a processor to dynamically update a trace cache with up-to-date information regarding the outcome of branches within a trace. Said updating would allow a processor to maintain an accurate representation of the most recent performance of a given trace. This advantage is desirable in the environment provided by Rotenberg as it would have allowed for fewer mispredictions of branch instructions while enabling a processor to maintain the use of a trace cache architecture. This advantage would have motivated one of ordinary skill in the art to allow a trace cache to be updated upon an invalid branch prediction of an instruction within a trace.

7. Claims 9-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg in view of Nair and in view of Patel and in view of Tanenbaum.

Regarding **claim 9**, Rotenberg, Nair and Tanenbaum disclose the limitations as stated in **claim 8**.

Rotenberg, Nair and Tanenbaum do not disclose further comprising determining based on which instruction within a block of instructions creates the branch whether the new trace is generated.

Patel does disclose further comprising determining based on which instruction within a block of instructions creates the branch whether the new trace is generated [see Patel, Page. 75, lines 7-9].

The advantage of determining whether a new trace should be generated based on its location within a block of instructions would have been to minimize the negative effects of trace fragmentation within the trace cache [see Patel, Page 75, lines 12-13], for example, in tight loop situations utilizing backwards branch instructions [see Patel, Page 75, lines 13-14]. This advantage is desirable in the environment disclosed by Rotenberg and Nair as it would have increased the throughput of the processor and the speed at which instructions could be fetched due to the minimization of fragmentation within the trace cache. This advantage would have motivated one of ordinary skill in the art at the time of invention to base the generation of new traces on the position of an instruction, as disclosed by Patel, within the invention disclosed by Rotenberg and Patel.

Regarding **claim 10**, Rotenberg, Nair and Tanenbaum disclose the limitations as stated in **claim 8**.

Rotenberg, Nair and Tanenbaum do not explicitly disclose *the alternate trace* [being] *generated* [being] *based on which block of instructions the branch occurs in.*

However, Patel discloses selectively packing traces dependent on their position within a trace block. Patel does not explicitly disclose selecting traces to be packed based on block position, however, Patel states the goal of selectively generating traces based on instruction position would have been to increase the fetch bandwidth of a trace cache. It would have been obvious to one of ordinary skill in the art at the time of invention to base the generation of a trace based on the block the trace resides in as it is merely a higher level of hierarchy (instruction – trace – block – cache) and the implementation would have been theoretically similar to that of denying the creation of a trace based on the instruction placement within a block. The advantages of selectively generating traces has been discussed in the arguments concerning the preceding claim and will not be repeated in this rejection. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to utilize the trace creation selection mechanism disclosed by Patel in a higher level of cache hierarchy, such as the block level.

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within

TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Vincent Lai whose telephone number is (571) 272-6749. The examiner can normally be reached on M-F 8:00-5:30 (First BiWeek Friday Off).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Alford Kindred can be reached on (571) 272-4037. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Vincent Lai
Examiner
Art Unit 2181

vl
July 12, 2007

7/19/2007

ALFORD KINDRED
PRIMARY EXAMINER